Subscribe (Full Service)   Register (Limited Service, Free)   Login

Search: ◉ The ACM Digital Library   ○ The Guide

"type propagation" +call +graph +direct +virtual   **SEARCH**

THE ACM DIGITAL LIBRARY

ℹ️ Feedback Report a problem Satisfaction survey

Terms used **type propagation** **call** **graph** **direct** **virtual**

Found **23** of **65 searched out of 65**.

Sort results by   [relevance ▽]

Display results   [expanded form ▽]

❧ Save results to a Binder

❓ Search Tips

☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 23

Result page: **1**   2   next

Relevance scale ☐ ◩ ◪ ◨ ■

1 Practical virtual method call resolution for Java   ■

Vijay Sundaresan, Laurie Hendren, Chrislain Razafimahefa, Raja Vallée-Rai, Patrick Lam, Etienne Gagnon, Charles Godin

October 2000 **ACM SIGPLAN Notices , Proceedings of the 15th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications,** Volume 35 Issue 10

Full text available: 📄 pdf(323.98 KB)   Additional Information: full citation, abstract, references, citings, index terms

This paper addresses the problem of resolving virtual method and interface calls in Java bytecode. The main focus is on a new practical technique that can be used to analyze large applications. Our fundamental design goal was to develop a technique that can be solved with only one iteration, and thus scales linearly with the size of the program, while at the same time providing more accurate results than two popular existing linear techniques, *class hierarchy analysis* and *rapid type an* ...

2 Using types to analyze and optimize object-oriented programs   ■

Amer Diwan, Kathryn S. McKinley, J. Eliot B. Moss

January 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS),** Volume 23 Issue 1

Full text available: 📄 pdf(414.51 KB)   Additional Information: full citation, abstract, references, citings, index terms

Object-oriented programming languages provide many software engineering benefits, but these often come at a performance cost. Object-oriented programs make extensive use of method invocations and pointer dereferences, both of which are potentially costly on modern machines. We show how to use types to produce effective, yet simple, techniques that reduce the costs of these features in Modula-3, a statically typed, object-oriented language. Our compiler performs type-based alias analysis to ...

Keywords: alias analysis, classes and objects, method invocation, object orientation, polymorphism, redundancy elimination

3 Simple and effective analysis of statically-typed object-oriented programs   ■

Amer Diwan, J. Eliot B. Moss, Kathryn S. McKinley

October 1996 **ACM SIGPLAN Notices , Proceedings of the 11th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications,** Volume 31 Issue 10

Full text available: 📄 pdf(1.70 MB)   Additional Information: full citation, abstract, references, citings, index terms

To use modern hardware effectively, compilers need extensive control-flow information.

Unfortunately, the frequent method invocations in object-oriented languages obscure control flow. In this paper, we describe and evaluate a range of analysis techniques to convert method invocations into direct calls for statically-typed object-oriented languages and thus improve control-flow information in object-oriented languages. We present simple algorithms for *type hierarchy analysis, aggregate analys ...*

4 SSA-based flow-sensitive type analysis: combining constant and type propagation

Alexandre Lenart, Christopher Sadler, Sandeep K. S. Gupta
March 2000 **Proceedings of the 2000 ACM symposium on Applied computing**

Full text available: pdf(387.03 KB)   Additional Information: full citation, references, citings, index terms

Keywords: SSA-form, object-oriented language, type analysis

5 A comparative study of static and profile-based heuristics for inlining

Matthew Arnold, Stephen Fink, Vivek Sarkar, Peter F. Sweeney
January 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN workshop on Dynamic and adaptive compilation and optimization**, Volume 35 Issue 7

Full text available: pdf(1.13 MB)   Additional Information: full citation, abstract, references, citings, index terms

In this paper, we present a comparative study of static and profile-based heuristics for inlining. Our motivation for this study is to use the results to design the best inlining algorithm that we can for the Jalapeño dynamic optimizing compiler for Java [6]. We use a well-known approximation algorithm for the KNAPSACK problem as a common "meta-algorithm" for the inlining heuristics studied in this paper. We present performance results for an implementation of these inlinin ...

6 Dynamic optimistic interprocedural analysis: a framework and an application

Igor Pechtchanski, Vivek Sarkar
October 2001 **ACM SIGPLAN Notices , Proceedings of the 16th ACM SIGPLAN conference on Object oriented programming, systems, languages, and applications**, Volume 36 Issue 11

Full text available: pdf(1.78 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

In this paper, we address the problem of *dynamic optimistic interprocedural analysis.* Our goal is to build on past work on *static interprocedural analysis* and *dynamic optimization* by combining their advantages. We present a framework for performing dynamic optimistic interprocedural analysis. the framework is designed to be used in the context of dynamic class loading and dynamic compilation, and includes mechanisms for event notification (on class loading and method compila ...

7 Static and dynamic analysis of call chains in java

Atanas Rountev, Scott Kagan, Michael Gibas
July 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 2004 ACM SIGSOFT international symposium on Software testing and analysis**, Volume 29 Issue 4

Full text available: pdf(276.86 KB)   Additional Information: full citation, abstract, references, index terms

This work presents a parameterized framework for static and dynamic analysis of call chains in Java components. Such analyses have a wide range of uses in tools for software understanding and testing. We also describe a test coverage tool built with these analyses and the use of the tool on a real-world test suite. Our experiments evaluate the exact precision of several instances of the framework and provide a novel approach for estimating the limits of class analysis technology for computing pr ...

Keywords: call chains, call graph, dynamic analysis, static analysis

8 Applying predication to efficiently handle runtime class testing

Chris Sadler, Sandeep K. S. Gupta, Rohit Bhatia

March 2000 **ACM SIGARCH Computer Architecture News**, Volume 28 Issue 1

Full text available: pdf(741.25 KB)   Additional Information: full citation, abstract, index terms

> Runtime class testing is a technique whereby virtual function calls are transformed into statically-bound function calls through a series of conditional branches. Through this transformation, the overhead of virtual function calls can be significantly reduced. However, the drawback of these tests is that by relying on conditional branches, the amount of instruction-level parallelism (ILP) is limited and the mispredict penalties can be relatively high. We show that by using predication during cla ...

9 Field analysis: getting useful and low-cost interprocedural information

Sanjay Ghemawat, Keith H. Randall, Daniel J. Scales

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5

Full text available: pdf(686.96 KB)   Additional Information: full citation, abstract, references, citings, index terms

> We present a new limited form of interprocedural analysis called field analysis that can be used by a compiler to reduce the costs of modern language features such as object-oriented programming, automatic memory management, and run-time checks required for type safety. Unlike many previous interprocedural analyses, our analysis is cheap, and does not require access to the entire program. Field analysis exploits the declared access restrictions placed on fields in a modul ...

10 Practicing JUDO: Java under dynamic optimizations

Michał Cierniak, Guei-Yuan Lueh, James M. Stichnoth

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5

Full text available: pdf(190.06 KB)   Additional Information: full citation, abstract, references, citings, index terms

> A high-performance implementation of a Java Virtual Machine (JVM) consists of efficient implementation of Just-In-Time (JIT) compilation, exception handling, synchronization mechanism, and garbage collection (GC). These components are tightly coupled to achieve high performance. In this paper, we present some static and dynamic techniques implemented in the JIT compilation and exception handling of the Microprocessor Research Lab Virtual Machine (MRL VM), ...

11 Student competition papers: A high-level view of Java applications

Eric Bodden

October 2003 **Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**

Full text available: pdf(72.78 KB)   Additional Information: full citation, abstract, references, index terms

> Static analysis of object-oriented applications has become widespread over the last decade, mainly in the context of compile-time optimizations. The paper describes how static analysis of virtual method calls can be employed to provide a high-level view of Java applications. The result is a method call graph that can be built from either source or bytecode, and a graphical browser that enables the user to analyze control flow and the coupling between classes and packages in an intuitive fashion, ...

> **Keywords**: Java, call graph visualization, static analysis

12 Effective synchronization removal for Java

Erik Ruf

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference**

on Programming language design and implementation, Volume 35 Issue 5

Full text available: pdf(819.53 KB)   Additional Information: full citation, abstract, references, citings, index terms

We present a new technique for removing unnecessary synchronization operations from statically compiled Java programs. Our approach improves upon current efforts based on escape analysis, as it can eliminate synchronization operations even on objects that escape their allocating threads. It makes use of a compact, equivalence-class-based representation that eliminates the need for fixed point operations during the analysis. We describe and evaluate the performance of an implemen ...

13 Strength reduction for loop-invariant types

Phung Hua Nguyen, Jingling Xue

January 2004 **Proceedings of the 27th conference on Australasian computer science - Volume 26**

Full text available: pdf(147.42 KB)   Additional Information: full citation, abstract, references

Types are fundamental for enforcing levels of abstraction in modern high-level programming languages and their lower-level representations. However, some type-related features such as dynamic method calls and dynamic type casts can contribute substantially to the performance of a program. Loop-invariant type is a concept relating to an object whose dynamic type never changes inside a loop. In this case, operations on the type of the object may be redundant in the loop. As these operations often ...

Keywords: PRE, inlining, loop-invariant type, strength reduction, type checking

14 Type feedback vs. concrete type inference: a comparison of optimization techniques for object-oriented languages

Ole Agesen, Urs Hölzle

October 1995 **ACM SIGPLAN Notices , Proceedings of the tenth annual conference on Object-oriented programming systems, languages, and applications**, Volume 30 Issue 10

Full text available: pdf(2.27 MB)   Additional Information: full citation, abstract, references, citings, index terms

Two promising optimization techniques for object-oriented languages are type feedback (profile-based receiver class prediction) and concrete type inference (static analysis). We directly compare the two techniques, evaluating their effectiveness on a suite of 23 SELF programs while keeping other factors constant.Our results show that both systems inline over 95% of all sends and deliver similar overall performance with one exception: SELF's automatic coercion of machine integer ...

15 Practical experience with an application extractor for Java

Frank Tip, Chris Laffra, Peter F. Sweeney, David Streeter

October 1999 **ACM SIGPLAN Notices , Proceedings of the 14th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications**, Volume 34 Issue 10

Full text available: pdf(2.31 MB)   Additional Information: full citation, abstract, references, citings, index terms

Java programs are routinely transmitted over low-bandwidth network connections as compressed class file archives (i.e., zip files and jar files). Since archive size is directly proportional to download time, it is desirable for applications to be as small as possible. This paper is concerned with the use of program transformations such as removal of dead methods and fields, inlining of method calls, and simplification of the class hierarchy for reducing application size. Such "extract ...

16 Efficient and precise modeling of exceptions for the analysis of Java programs

Jong-Deok Choi, David Grove, Michael Hind, Vivek Sarkar

September 1999 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 1999**

ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering, Volume 24 Issue 5

Full text available: 📄 pdf(1.16 MB)   Additional Information: full citation, abstract, references, citings, index terms

The Factored Control Flow Graph, *FCFG,* is a novel representation of a program's intraprocedural control flow, which is designed to efficiently support the analysis of programs written in languages, such as Java, that have frequently occurring operations whose execution may result in exceptional control flow. The FCFG is more compact than traditional CFG representations for exceptional control flow, yet there is no loss of precision in using the FCFG. In this paper, we introduce the FCFG r ...

17 ABCD: eliminating array bounds checks on demand

Rastislav Bodík, Rajiv Gupta, Vivek Sarkar

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5

Full text available: 📄 pdf(306.96 KB)   Additional Information: full citation, abstract, references, citings, index terms

To guarantee typesafe execution, Java and other strongly typed languages require bounds checking of array accesses. Because array-bounds checks may raise exceptions, they block code motion of instructions with side effects, thus preventing many useful code optimizations, such as partial redundancy elimination or instruction scheduling of memory operations. Furthermore, because it is not expressible at bytecode level, the elimination of bounds checks can only be performed at run time ...

18 A model of authorization for next-generation database systems

Fausto Rabitti, Elisa Bertino, Won Kim, Darrell Woelk

March 1991 **ACM Transactions on Database Systems (TODS)**, Volume 16 Issue 1

Full text available: 📄 pdf(2.79 MB)   Additional Information: full citation, abstract, references, citings, index terms, review

The conventional models of authorization have been designed for database systems supporting the hierarchical, network, and relational models of data. However, these models are not adequate for next-generation database systems that support richer data models that include object-oriented concepts and semantic data modeling concepts. Rabitti, Woelk, and Kim [14] presented a preliminary model of authorization for use as the basis of an authorization mechanism in such database systems. In this p ...

Keywords: object-oriented database, semantic database

19 Safety checking of machine code

Zhichen Xu, Barton P. Miller, Thomas Reps

May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation**, Volume 35 Issue 5

Full text available: 📄 pdf(306.71 KB)   Additional Information: full citation, abstract, references, citings, index terms

We show how to determine statically whether it is safe for untrusted machine code to be loaded into a trusted host system. Our safety-checking technique operates directly on the untrusted machine-code program, requiring only that the initial inputs to the untrusted program be annotated with typestate information and linear constraints. This approach opens up the possibility of being able to certify code produced by any compiler from any source language, which gives the code prod ...

20 An Automatic Technique for Selection of Data Representations in SETL Programs

Edmond Schonberg, Jacob T. Schwartz, Micha Sharir

April 1981 **ACM Transactions on Programming Languages and Systems (TOPLAS),** Volume 3 Issue 2

Full text available: 📄 pdf(1.22 MB)   Additional Information: full citation, references, citings, index terms

Results 1 - 20 of 23   Result page: **1**   2   next

Useful downloads: Adobe Acrobat   QuickTime   Windows Media Player   Real Player

P✺RTAL

US Patent & Trademark Office

Search: ◉ The ACM Digital Library   ○ The Guide

"type propagation" +call +graph +direct +virtual   SEARCH

THE ACM DIGITAL LIBRARY

❧ Feedback Report a problem Satisfaction survey

Terms used **type propagation** **call** **graph** **direct** **virtual**

Found **23** of **65**

Sort results by   relevance ▼

Display results   expanded form ▼

❧ Save results to a Binder

? Search Tips

☐ Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 21 - 23 of 23   Result page: previous   1   **2**

Relevance scale ☐☐▬▬■

21 Design of an optimizing, dynamically retargetable compiler for common Lisp   ▬

Rodney A. Brooks, David B. Posner, James L. McDonald, Jon L. White, Eric Benson, Richard P. Gabriel

August 1986 **Proceedings of the 1986 ACM conference on LISP and functional programming**

Full text available: 📄 pdf(1.13 MB)   Additional Information: full citation, references, citings

22 Flexible support for multiple access control policies   ▬

Sushil Jajodia, Pierangela Samarati, Maria Luisa Sapino, V. S. Subrahmanian

June 2001 **ACM Transactions on Database Systems (TODS)**, Volume 26 Issue 2

Full text available: 📄 pdf(460.33 KB)   Additional Information: full citation, abstract, references, citings, index terms

Although several access control policies can be devised for controlling access to information, all existing authorization models, and the corresponding enforcement mechanisms, are based on a specific policy (usually the closed policy). As a consequence, although different policy choices are possible in theory, in practice only a specific policy can actually be applied within a given system. In this paper, we present a unified framework that can enforce multiple access control policies withi ...

Keywords: access control policy, authorization, logic programming

23 A fine-grained access control system for XML documents   ▬

Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, Pierangela Samarati

May 2002 **ACM Transactions on Information and System Security (TISSEC)**, Volume 5 Issue 2

Full text available: 📄 pdf(330.60 KB)   Additional Information: full citation, abstract, references, citings, index terms

Web-based applications greatly increase information availability and ease of access, which is optimal for public information. The distribution and sharing of information via the Web that must be accessed in a selective way, such as electronic commerce transactions, require the definition and enforcement of security controls, ensuring that information will be accessible only to authorized entities. Different approaches have been proposed that address the problem of protecting information in a Web ...

Keywords: Access control, World Wide Web, XML documents, authorizations specification and enforcement

IEEE *Xplore* ©
RELEASE 2.0

Welcome United States Patent and Trademark Office

☐ Search Results

BROWSE   SEARCH   IEEE XPLORE GUIDE   SUPPORT

Results for "((type propagation)<in>metadata)"
Your search matched **10** of **1152881** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

not relevant

☑ e-mail 🖨 printer friendly

» <u>View Session History</u>

» <u>New Search</u>

» **Key**

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

Modify Search

((type propagation)<in>metadata)   [»]

☐ Check to search only within this results set

Display Format: ◉ Citation   ○ Citation & Abstract

Select   **Article Information**

☐   1. **Propagation of electromagnetic waves in cylindrical anisotropic dielectric waveguide**
Bezugly, A.V.; Khoroshun, V.V.;
Direct and Inverse Problems of Electromagnetic and Acoustic Wave Theory, 1997. DIPED-97
15-17 Sept. 1997 Page(s):51

<u>AbstractPlus</u> | Full Text: <u>PDF</u>(44 KB)   IEEE CNF

☐   2. **Bearing estimation for a distributed source: modeling, inherent accuracy limitations and algorithms**
Raich, R.; Goldberg, J.; Messer, H.;
Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]
Volume 48, Issue 2, Feb. 2000 Page(s):429 - 441

<u>AbstractPlus</u> | <u>References</u> | Full Text: <u>PDF</u>(372 KB)   IEEE JNL

☐   3. **A new method for quantifying the motion of epicardial activation wavefronts**
Rogers, J.M.; Ideker, R.E.; Smith, W.M.;
Computers in Cardiology 1995
10-13 Sept. 1995 Page(s):797 - 800

<u>AbstractPlus</u> | Full Text: <u>PDF</u>(328 KB)   IEEE CNF

☐   4. **Space-time equalization for DVB-T in single frequency networks**
Magniez, P.; Gorokhov, A.;
Signals, Systems, and Computers, 1999. Conference Record of the Thirty-Third Asilomar Conference on
Volume 1, 24-27 Oct. 1999 Page(s):622 - 626 vol.1

<u>AbstractPlus</u> | Full Text: <u>PDF</u>(404 KB)   IEEE CNF

☐   5. **Combining directional channel modelling with beamforming adaptive antennas for UMTS**
Gil, J.M.; Correia, L.M.;
Personal, Indoor and Mobile Radio Communications, 2001 12th IEEE International Symposium on
Volume 2, 30 Sept.-3 Oct. 2001 Page(s):E-132 - E-136 vol.2

<u>AbstractPlus</u> | Full Text: <u>PDF</u>(444 KB)   IEEE CNF

☐   6. **The propagation of electromagnetic waves in ionized gases**
Northover, F.;
Antennas and Propagation, IEEE Transactions on [legacy, pre - 1988]
Volume 7, Issue 5, Dec 1959 Page(s):340 - 360

<u>AbstractPlus</u> | Full Text: <u>PDF</u>(1608 KB)   IEEE JNL

☐   7. **Radiation and guided waves**
Karbowiak, A.;
Antennas and Propagation, IEEE Transactions on [legacy, pre - 1988]
Volume 7, Issue 5, Dec 1959 Page(s):191 - 200

☐   8.   **Spatial Properties of Meteor-Burst Propagation**
Ince, A.;
Communications, IEEE Transactions on [legacy, pre - 1988]
Volume 28, Issue 6, Jun 1980 Page(s):841 - 849

AbstractPlus | Full Text: PDF(1104 KB)   IEEE JNL

☐   9.   **Interference calculations for an S-CDMA cellular radio system**
Tanik, Y.; Kuzuoglu, M.;
Electrotechnical Conference, 1994. Proceedings., 7th Mediterranean
12-14 April 1994 Page(s):121 - 124 vol.1

AbstractPlus | Full Text: PDF(280 KB)   IEEE CNF

☐   10.   **Radar-type propagation survey experiments for communication systems**
Lacy, R.; Sharp, C.;
IRE International Convention Record
Volume 4, Mar 1956 Page(s):20 - 27

AbstractPlus | Full Text: PDF(1024 KB)   IEEE CNF

US005428793A

United States Patent [19]

Odnert et al.

[11] Patent Number: 5,428,793

[45] Date of Patent: Jun. 27, 1995

[54] **METHOD AND APPARATUS FOR COMPILING COMPUTER PROGRAMS WITH INTERPROCEDUURAL REGISTER ALLOCATION**

[75] Inventors: **Daryl Odnert**, Boulder Creek; **Vatsa Santhanam**, Sunnyvale, both of Calif.

[73] Assignee: **Hewlett-Packard Company**, Palo Alto, Calif.

[21] Appl. No.: **435,914**

[22] Filed: **Nov. 13, 1989**

[51] Int. Cl.⁶ G06F 9/44; G06F 9/45

[52] U.S. Cl. 395/700; 364/232.23; 364/280.5; 364/DIG. 1

[58] Field of Search ... 364/200 MS File, 900 MS File; 395/700, 375

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,571,678	2/1986	Chaitin	364/200
4,642,764	2/1987	Auslander et al.	395/700
4,656,582	4/1987	Chaitin et al.	364/200
4,777,588	10/1988	Case et al.	395/800
4,782,444	11/1988	Munshi et al.	364/200
4,953,084	8/1990	Meloy et al.	364/200
4,953,106	8/1990	Gansner et al.	364/521
4,961,141	10/1990	Hopkins et al.	364/200
5,083,263	1/1992	Joy et al.	395/425
5,161,216	11/1992	Reps et al.	395/375

OTHER PUBLICATIONS

Wall, "Global Register Allocation at Link Time", Proc. of the SIGPLAN, SIGPLAN Notices, vol. 21, No. 7, Jul. 1986, pp. 264–275.

Fred C. Chow, "Minimizing Register Usage Penalty at Procedure Calls", *Proceedings of the SIGPLAN*, '88 Conference on Programming Language Design and Implementation, Jul. 1988, pp. 85–94.

Mark I. Himelstein, et al., "Cross-Module Optimiza-

tions: Its Implementation and Benefits", *Proceedings of the Summer 1987 USENIX Conference*, pp. 347–356.

P. A. Steenkiste, "LISP on a Reduced Instruction Set Processor: Characterization and Optimization", Stanford University Computer Systems Laboratory, PhD Thesis, Chapter 5, Mar. 1987.

Hans Mulder, "Data Buffering: Run–Time Versus Compile Time Support", *Proceedings of the 3rd International Conference on Architectural Support for Programming Languages and Operating Systems*, Apr. 3–6, 1989, pp. 144–151.

Keith D. Cooper, et al., "The Impact of Interprocedural Analysis and Optimization in R^n Programming Environment", Published in ACM Transactions on Programming Languages and Systems, Oct. 1986, pp. 491–523.

Primary Examiner—Kevin A. Kriess
Assistant Examiner—D. Butler

[57] **ABSTRACT**

Optimization techniques are implemented by means of a program analyzer used in connection with a program compiler to optimize usage of limited register resources in a computer processor. The first optimization technique, called interprocedural global variable promotion allows the global variables of a program to be accessed in common registers across a plurality of procedures. Moreover, a single common register can be used for different global variables in distinct regions of a program call graph. This is realized by identifying subgraphs, of the program call graph, called webs, where the variable is used. The second optimization technique, called spill code motion, involves the identification of regions of the call graph, called clusters, that facilitate the movement of spill instructions to procedures which are executed relatively less often. This decreases the overhead of register saves and restores which must be executed for procedure calls.

4 Claims, 7 Drawing Sheets